Abstract

A method for developing an automation client program in a graphical programming environment is disclosed. The graphical programming environment provides a set of automation nodes and controls which may be dropped and wired together to create a graphical program. The nodes include an automation refnum which references a user-selected automation class from an automation type library exported by an automation server application, such as Microsoft Excel; an automation open node which instantiates an object from the selected automation class; an automation invoke node which invokes a user-selected method of the automation class; and an automation property node which invokes, i.e., reads or writes, user-selected properties of the automation class. The nodes enable the displaying, manipulating, cataloging, editing or performance other operations, such as may be performed by an automation server, on data acquired or generated by a virtual instrument. A method for performing class propagation and type propagation checking of automation objects in a graphical program is also disclosed. The automation class of a first automation node is propagated from the first node to a second automation node when the two nodes are wired together or when the automation class of the first node is changed to a second class. The automation invoke node and automation property node perform type checking to verify that the user-selected method or property is valid for, i.e., defined by, the automation class of the node. The node requests an object manager to determine whether or not the method or property is valid. The object manager queries a type library which the automation class is in, in order to obtain a list of valid methods and properties for the automation class. The object manager searches the list to determine if the specified method or property is present in the list, i.e., is valid.